Hindawi
Complexity
Volume 2024, Article ID 9837865, 1 page
https://doi.org/10.1155/2024/9837865

Retraction

Retracted: Knowledge Graph Entity Similarity Calculation under Active Learning

Complexity

Received 23 January 2024; Accepted 23 January 2024; Published 24 January 2024

This article has been retracted by Hindawi following an investigation undertaken by the publisher [1]. This investigation has uncovered evidence of one or more of the following indicators of systematic manipulation of the publication process:

(1) Discrepancies in scope

(2) Discrepancies in the description of the research reported

(3) Discrepancies between the availability of data and the research described

(4) Inappropriate citations

(5) Incoherent, meaningless and/or irrelevant content included in the article

(6) Manipulated or compromised peer review

The presence of these indicators undermines our confidence in the integrity of the article's content and we cannot, therefore, vouch for its reliability. Please note that this notice is intended solely to alert readers that the content of this article is unreliable. We have not investigated whether authors were aware of or involved in the systematic manipulation of the publication process.

Wiley and Hindawi regrets that the usual quality checks did not identify these issues before publication and have since put additional measures in place to safeguard research integrity.

We wish to credit our own Research Integrity and Research Publishing teams and anonymous and named external researchers and research integrity experts for contributing to this investigation.

The corresponding author, as the representative of all authors, has been given the opportunity to register their agreement or disagreement to this retraction. We have kept a record of any response received.

References

[1] L. Li, Z. Zhang, and S. Zhang, "Knowledge Graph Entity Similarity Calculation under Active Learning," *Complexity*, vol. 2021, Article ID 3522609, 11 pages, 2021.

Hindawi
Complexity
Volume 2021, Article ID 3522609, 11 pages
https://doi.org/10.1155/2021/3522609

Research Article

Knowledge Graph Entity Similarity Calculation under Active Learning

Lianhuan Li,[1] **Zheng Zhang**,[2] **and Shaoda Zhang**[3]

[1]*School of International Education, Nanyang Medical College, Nanyang, Henan 473000, China*
[2]*School of Computer and Software, Nanyang Institute of Technology, Nanyang, Henan 473000, China*
[3]*Cofoe Medical Technology Company Limited, Shenzhen, Guangdong 518101, China*

Correspondence should be addressed to Zheng Zhang; zhangzheng@nyist.edu.cn and Shaoda Zhang; zhangsd@stu.xmu.edu.cn

Received 25 April 2021; Revised 20 May 2021; Accepted 1 June 2021; Published 12 June 2021

Academic Editor: Zhihan Lv

To address the objectives of the adaptive learning platform, the requirements of the system in terms of business, functionality, and performance are mainly analysed, and the design of functions and database is completed; then, an updatable learner model is constructed based on the cognitive diagnosis model and resource preference attributes; then, the construction of the knowledge map is completed based on embedding to achieve knowledge point alignment, and based on this, the target knowledge points of learners are located with the help of deep learning; at the same time, the target knowledge points are taken as the starting point to generate the best learning path by traversing the knowledge map, and the corresponding learning resources and test questions are recommended for them with the help of the architecture; finally, the adaptive learning platform is developed in the environment using the architecture. Also, the target knowledge point is used as the starting point to traverse the knowledge map to generate the best learning path, and the corresponding learning resources and test questions are recommended for the learner in combination with the learner model; finally, this study adopts an architecture for the development of an adaptive learning platform in the environment to realize online tests, score analysis, resource recommendation, and other functions. A knowledge graph fusion system supporting interactive facilitation between entity alignment and attribute alignment is implemented. Under a unified conceptual layer, this system can combine entity alignment and attribute alignment to promote each other and truly achieve the final fusion of the two graphs. Our experimental results on real datasets show that the entity alignment algorithm proposed in this paper has a great improvement in accuracy compared with the previous mainstream alignment algorithms. Also, the attribute alignment algorithm proposed in this paper, which calculates the similarity based on associated entities, outperforms the traditional methods in terms of accuracy and recall.

1. Introduction

In today's era of high-speed Internet development and explosive growth of information, people are prone to problems such as information overload, which makes it difficult to obtain effective information and learn knowledge. Also, many times users' goals are not clear and prone to information disorientation [1]. To solve such problems, recommendation systems have attracted the attention of many experts and scholars. Recommender systems are mainly used to show users the products or services they want automatically and timely through the information filtering of recommendation algorithms. Although a search engine is also a technology that can provide information filtering services for users, its performance is far inferior to that of a recommendation system when users' needs are not clear or cannot be accurately described by keywords. In the era of mobile Internet, many products are containing such application scenarios; for example, our common APPs of news and information, e-commerce, music, etc. all need to use recommendation systems for content distribution [2]. Compared with manual orchestration recommendations, the advantage of the recommendation system is that there are thousands of people, fully automated, and high update efficiency. Therefore, it is easier for a personalized recommendation system to hit users' interest points, thus

enhancing user experience, promoting user click and purchase conversion, and generating business value.

At present, there are many methods to solve the sparsity problem, and the common ones are simple to value filling, clustering, dimensionality reduction, and recommendation methods of fused content. Among them, the first two cannot reflect the difference between users' preferences and are not suitable for personalized recommendation scenarios; dimensionality reduction methods reduce the data by decreasing the dimensionality of item data, to alleviate the data sparsity problem, and the commonly used methods are principal component analysis and singular value decomposition [3]. Although the dimensionality reduction method can reduce the size and sparsity of the interaction matrix to a certain extent, it also loses part of the rating data and makes it difficult to guarantee the recommendation effect; finally, the recommended method of fused content solves the interaction matrix sparsity problem by introducing additional items or users' auxiliary information, such as representation vectors or display labels, to help to mine the users or items with similar nearest neighbors. Currently, the most representative information data which can be used at scale is the knowledge graph. To improve the effectiveness of recommendation algorithms, this paper will study how to combine the knowledge graph into personalized recommendation techniques [4]. To solve the problem that the traditional graph distributed representation method loses the higher-order similarity at the subgraph level, this paper proposes a recurrent neural network-based knowledge graph distributed representation model KG-GRU, which models the subgraph similarity using the sequence containing nodes and relations and represents the relations and nodes in the same embedding vector space. Also, this paper proposes a skip or stay strategy JUST to guide random wandering for data sampling of the knowledge graph, avoiding the problems of manually constructing meta-paths and an unbalanced distribution of node types. Finally, this paper demonstrates through link prediction experiments that the KG-GRU model can learn more accurate distributed representation vectors of entities and relations in the film domain mapping constructed in this paper [5].

The same sparsity exists for entities and relationships in knowledge graphs. In statistics, there is a relatively common phenomenon of the long tail of data, and there are many methods in statistics to solve the problem. The main reason is that certain entities and relationships appear very frequently, such as celebrities or political figures, because there are more articles or reports related to them, and likewise, there are many relationships related to them, while another kind of entities and relationships appears less frequently, such as an unknown person, which appears not only less frequently but also in large numbers. Based on the idea of a collaborative filtering algorithm with fused contents, KG-CF directly fuses the distributed representation vector of items in the knowledge graph of the movie domain into the item similarity calculation; i.e., it supplements the semantic information of items to the traditional item-based collaborative filtering algorithm and thus improves the personalized recommendation effect. KG-GRU4Rec improves on the distributed representation model of knowledge graph proposed in this paper, KG- GRU, an end-to-end model for predicting users' ratings of movies which is implemented, avoiding the problem that the rating prediction of KG-CF still relies on users' historical rating data. Finally, this paper demonstrates that the proposed KG-GRU4Rec recommendation algorithm outperforms the comparison algorithms in terms of hit rate and average backward ranking through Top-N movie recommendation experiments.

2. Status of Research

This leads to the sparse entities and relations related to them, and this problem becomes increasingly obvious especially when the size of the knowledge graph keeps getting larger. Few instances of attributes and relationships of long-tail entities lead to serious missing of attributes and relationships, which will seriously affect the effect of knowledge graph complementation.

Knowledge graphs are divided into generic knowledge graphs and industry knowledge graphs [6]. General knowledge graph data are mainly derived from Wikipedia and various domains and are constructed based on projects such as Linked Open Data (LOD). Industry knowledge graphs, also called vertical domain knowledge graphs, use data in a domain for graph construction, which is a description of a specific domain and therefore involves a narrower range of data, but a more accurate knowledge base. The data collection phase collects the needed data from the web [7]. Noy et al. designed a distributed collection method for topic-specific data, which uses a multistrategy fusion search algorithm to collect data in parallel while filtering the data according to the topic similarity algorithm to obtain topic-related data [8]. Amreen et al. designed and implemented a distributed crawler system, which can use a task editing interface [9]. Gaur et al. ported a single-threaded or multithreaded web crawler to a Hadoop distributed system to cope with the disadvantage that web crawlers cannot acquire data in a timely and effective manner in the Internet environment where a large amount of data exists in the presence of large amounts of data on the Internet and also to improve the scalability and reliability of the crawler [10].

Initially, concurrency techniques were used on single-node single processors for more efficient execution. Concurrency means that multiple tasks occur simultaneously at the same time interval; microscopically these tasks occupy the processor in time slices for execution and macroscopically these tasks occur simultaneously [11]. The limited computing power of single-node single-processor has given rise to multiprocessor or multinode-based parallel computing. Parallelism refers to the simultaneous execution of multiple tasks at the same moment, with each task occupying one processor or node for computation at the microscopic level. Multiple computer nodes collaborating to execute tasks involve task decomposition, internode communication, and other issues, so distributed computing frameworks and distributed message queues are generated [12]. For task-parallel computing, the use of distributed message queues has higher flexibility than distributed computing

frameworks, but it requires manual management of node resources to achieve task decomposition, distribution, and aggregation of computation results, so the development efficiency is lower. For parallel processing of deep learning, it is usually done using distributed deep learning frameworks due to the high complexity of these algorithms.

Message queues are often used as message middleware in distributed systems, which have the functions of reducing coupling between modules, enabling asynchronous communication between modules, and reducing system traffic peaks, etc. They are widely used in e-commerce systems, logging systems, and subscription publishing systems and are also very suitable in certain distributed computing tasks [13]. We analyse the process of knowledge graph construction and analyse the steps and algorithms that are not efficient for single-node processing in the construction process in conjunction with actual projects and select the applicable parallelization techniques to realize its parallelized processing and improve the efficiency of the algorithm without degrading its performance. We also design and implement the cluster node management platform and parallel algorithm startup interface to facilitate the use of parallel algorithms. A knowledge graph to accommodate all the entities and relationships in time is impossible to exist [14]. When the scope of a knowledge graph is limited, the fusion of new knowledge extracted from data sources such as web pages, text, XML, etc. into the existing knowledge graph is considered. However, there are various problems in the fusion process; for example, the same entity has different representations in different data sources, and different entities may have the same representation, etc. How to fuse multiple sources of data and combine existing knowledge graphs to do inference is an important research direction.

3. Analysis of Entity Similarity Calculation for Active Learning Knowledge Graphs

3.1. Design of Knowledge Graph Entity Similarity Calculation. There are mainly top-down, bottom-up, and a mixture of both approaches for knowledge graph construction [15]. The main difference between top-down and bottom-up construction is the different order of construction between the ontology layer, which represents concepts, and the entity layer, which stores instances: the top-up construction approach first constructs the ontology, which represents concepts, and then extracts entities from data to add to the knowledge base, while the bottom-up construction approach first extracts entities and relationships from data and then constructs the ontology layer. For the bottom-up approach, the main steps are knowledge extraction from various data, a fusion of structured or tripartite knowledge bases, knowledge processing such as knowledge inference, and validation, as shown in Figure 1. Knowledge extraction extracts the information needed to construct a mapping from unstructured and semistructured data and usually requires the extraction of entities, attributes, and relationships; the extracted data are organized in a certain way to form a knowledge base, and knowledge fusion is also involved if structured data and third-party knowledge bases are to be incorporated; the knowledge base obtained through the knowledge extraction and fusion stages have a large amount of redundant and erroneous information. There exist incomplete information, flattened data relationships, and lack of hierarchy and logic, and this requires knowledge processing processes such as ontology construction, knowledge reasoning, and quality assessment.

This method first constructs an ontology library based on the domain to form an ontology layer and then uses knowledge extraction technology to extract instances and relationships to form an entity layer, followed by the knowledge fusion and knowledge processing process. The hybrid approach starts with knowledge extraction and then forms the ontology layer and then fills in the entities with knowledge extraction based on the ontology layer and iteratively updates the entity layer and the ontology layer to form the knowledge graph. Humans usually know and relate the world through "events" and their related relationships, and it is difficult to represent events and interevent relationships in traditional knowledge graphs, so event-based knowledge graphs, also called matter graphs, are created [16]. Event knowledge is the core of the graph, and event elements are the detailed descriptions of events. Event knowledge graph construction has steps such as collecting event-related texts, extracting events and event elements from them, then analysing events and interevent relationships, and fusing redundant events. The currently existing knowledge graphs construct the association between entities, but the graphs lack relevant semantic information. To enable computers to understand chapters to a certain extent, a knowledge representation framework that represents chapter knowledge in terms of syntactic elements, syntactic element attributes, semantic elements, interelement relationships, and constraints is established. The construction process includes the steps of data acquisition, knowledge element extraction at each level, knowledge element representation, and element relationship discovery at each level.

The above briefly introduces the event knowledge graph and the chapter understanding knowledge graph. The construction techniques of the two knowledge graphs and the related single-node algorithms are completed by other students, and this paper focuses on analysing the parallelization methods of these algorithms and applying them. At the same time, due to the complexity of the knowledge graph construction process, there are many solutions to a certain problem in each stage, and it is not possible to cover them all. Therefore, we analyse the parallelization techniques of a specific algorithm in the knowledge graph construction process as an example for the parallelization of other similar algorithms for reference. The algorithm selection is mainly considered from three aspects, namely, practicability, generality, and time complexity: practicability refers to the special algorithm that needs parallelization in the actual construction of a specific knowledge graph; generality refers to the parallel processing method of the algorithm that can be applied to other similar algorithms with slight modifications; time complexity refers to the algorithm that has a large amount of data or a large amount of computation, and the processing time of a single node needs to be measured in weeks or months.

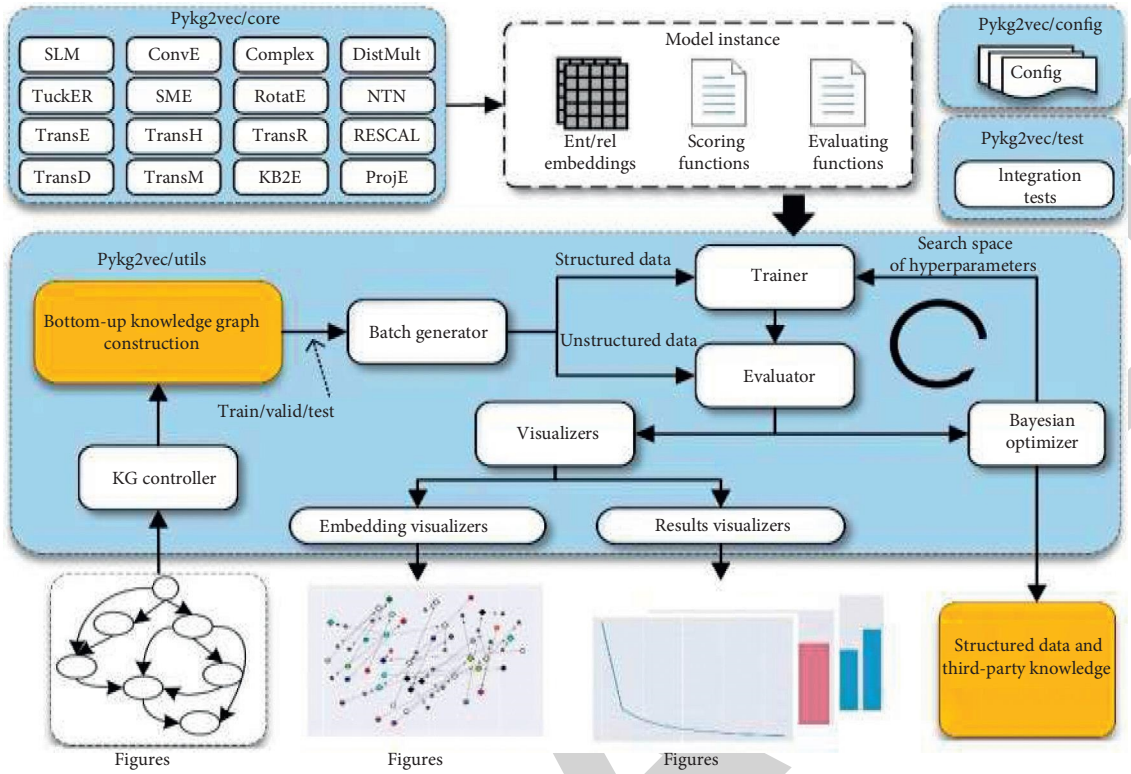

Figure 1: Bottom-up knowledge graph construction.

The subsequent experiments in this phase use the pulp tool as the basis for parallelized entity recognition, for which entity recognition can be viewed as a sequence labelling task: let $X = \{a_1, a_2, a_3, \ldots, a_n\}$, the random variable in the task, denote the sequence of observations to be labelled, and the random variable $Y = \{b_1, b_2, b_3, \ldots, b_n\}$ denote the labelled sequence corresponding to the sequence of observations; if the random variable Y satisfies the Markov independence assumption, then given the sequence X to be labelled, a particular labelled sequence Y has the probability of

$$P(y|x) = \frac{1}{Z\left(\sum_i \sum_i^n \lambda_j t_j\left(y_{i+1}, y_{i+1}, x, i, j\right)\right)}, \tag{1}$$

where $t_j(y_{i+1}, y_{i+1}, x, i, j)$ is the transfer characteristic function of two adjacent markers on the observed sequence, $S_k(y_{i+1}, i, j)$ is the state characteristic function on the marker position i of the observed sequence, y_{i+1} and u_k are parameters, and z are the normalization factor. In the task given the training set $D = \{X, Y\}$, where X is the observed data set and Y is the labelled data set, the model is trained using the data based on the maximum likelihood principle, and after taking the logarithm of the likelihood for the data set, the objective function is

$$\log L(\theta) = \sum_{i=1}^{n} p\left(x_i^2 y_i^2\right). \tag{2}$$

Given the model parameters, the most probable sequence of markers is found to be represented as

$$Y^* = \arg\min P(y|x)\exp\frac{1}{Z\left(\sum_i \sum_i^n \lambda_j t_j\left(y_{i+1}, y_{i+1}, x, i, j\right)\right)}. \tag{3}$$

The current position is i and the marker is y. The unnormalized probability value of the optimal sequence of markers to the current position is obtained using the Viterbi algorithm in the recursive form of

$$\theta(i, y) = \min y' \left\{\theta(i, y') \times l \sum_k \lambda_j t_j\left(y_{i+1}, y_{i+1}, x, i, j\right)\right\}. \tag{4}$$

The entities extracted from the text are fragmented and need to be further extracted from the corpus to find the associative relationships between the entities. Relationship extraction was mainly done by manually constructing syntactic and semantic rules in the early days, and due to poor scalability, methods based on statistical machine learning methods for modelling entities and relationships were generated, and semisupervised and unsupervised methods were subsequently generated to reduce manual annotation [17]. The process of entity and relationship extraction is not necessarily the same for different knowledge graphs, and extraction needs to be combined with various methods. The following are examples of event-oriented knowledge graphs and chapter-understanding-oriented knowledge graph entity and relationship extraction.

The multiple algorithms involved in this phase of entity and relationship extraction process have a common feature:

the algorithm input is a chapter or a single sentence, and if these data are processed under multiple nodes, each node processes different inputs without affecting each other, and the nodes do not involve internode communication when they perform their own data processing. For this type of algorithm, each input processing can be independent of each other and will not affect each other, so the data can be distributed to the cluster nodes and each node will process the data it gets assigned and the management node will just aggregate all the results. At the same time, the amount of data processed by each algorithm and the preprocessing before the algorithm runs may be different, so based on the above analysis three data parallelization methods are designed to cope with different situations; among the three parallelization methods, the first two are done using Spark framework, and the third one is done based on the distributed message queue.

The number of resources we assign to the head entity is 1; i.e., $R_p(h) = 1$. Recursively using the above equation, we can obtain the number of resources flowing from the head entity h to the tail entity t along with the path p, which is denoted as $R_p(t)$. The core idea of Phrase s is to learn not only the entities and relations in the knowledge graph as vectors but also the sequence of relations paths as vectors, as shown in Figure 2.

In the above process, using a convolution kernel $W = \{w_1, w_2, \ldots, w_n\}$ of length, the convolution operation can be expressed formally as follows:

$$c_i = \left\{c_{i,l} | c_{i,j} = w_i \otimes E_{(j+l+1):\ j}\right\}. \tag{5}$$

In turn, the semantic feature vector can be obtained as

$$C = \{c_1, c_2, \ldots, c_n\}. \tag{6}$$

From the above process, the fused item similarity finally used by the KG-CF recommendation algorithm in this section is derived from the item similarity of the user-item interaction matrix and the semantic similarity of the items based on the knowledge graph. However, in the face of natural utterances, RNN networks have some advantages over CNN networks in the sequence modelling process, due to their ability to capture the contextual information of the current moment in the sequence, by solving the long-range dependency problem. In the above DRNN network structure, the chain RNN network, which originally matches the whole sequence of sentences, is split into several RNN networks consisting of fixed-length recurrent units by setting a window. Accordingly, in the RNN network, the state at the current moment is related to all the historical states, while in the DRNN network, the output of the state at the current moment depends only on the historical states in the window in which it is located.

After the expert node starts, it needs to create a data queue, control queue, and result queue, read data from local or other data sources, add data to the data queue, and then listen to the result queue to get the results [18]. Each piece of data in the program corresponds to a result. The program determines whether the data is processed according to the number of results and sends an end message to the control queue to end the slave node after getting all the results. If data is obtained from the data queue, the named entity is identified using the identification object, and then the result is submitted to the result queue. If an end message is fetched from the control queue, the program ends, and other messages are fetched and processed accordingly.

3.2. Active Learning System Design. The platform is designed using *B*/*S* architecture to improve the compatibility of the system, using the front-end and back-end separation design approach to make the front-end rendering display have the advantage of front-end and back-end data interaction through JSON-web unified data format. Vue is a lightweight framework for writing front-end pages, front-end data rendering using MVVM, and real-time page refresh operations using the virtual DOM ideally to achieve two-way data binding capabilities [19]. Vue can easily achieve customization and modularity of functionality. At the same time, for some work such as data binding, routing configuration, project compilation Vue can also be easily competent. The logic diagram is shown in Figure 3. V-doom is responsible for page rendering and updating; v-router is responsible for jumping; state management object Vue is responsible for front-end data interaction, such as learner's rating of resources, passing test scores between components, etc.

The structure of the Vue model consists of a generator and a discriminator, and there is a minimal game between the generator and the discriminator like a minimal game between two game players. Through iterative training, the generator tricks the discriminator with the generated data by capturing the data distribution of the real samples as much as possible, while the discriminator distinguishes the generated data from the real data by judging the data it receives as true or false. In the early stage, Vue was used as a generative model because it was mostly used to generate various kinds of data in a continuous space. Essentially, the Vue model provides a learning framework in which initial unrefined data can be augmented by adversarial training between generators and discriminators.

The acquisition process of global and local semantic features is defined as the feature generation process, and the relevant models are collectively referred to as generators, while the discriminator consists of a two-layer fully connected neural network with ReLU used as its activation function. The semantic features extracted by the model are regarded as negative samples, while the positive samples are obtained by the manual precision representation of semantic relations based on the shortest dependency path analysis of the corresponding threat intelligence sentences and by performing adversarial training to reduce the difference between negative and positive samples in the distribution of potential features, i.e., to enhance the semantic feature representation extracted by the model.

For the model input containing only word features, the accuracy and recall of the model corresponding to the extraction results are relatively low, resulting in a relatively low overall performance of the model, with only 71.84% *F*1

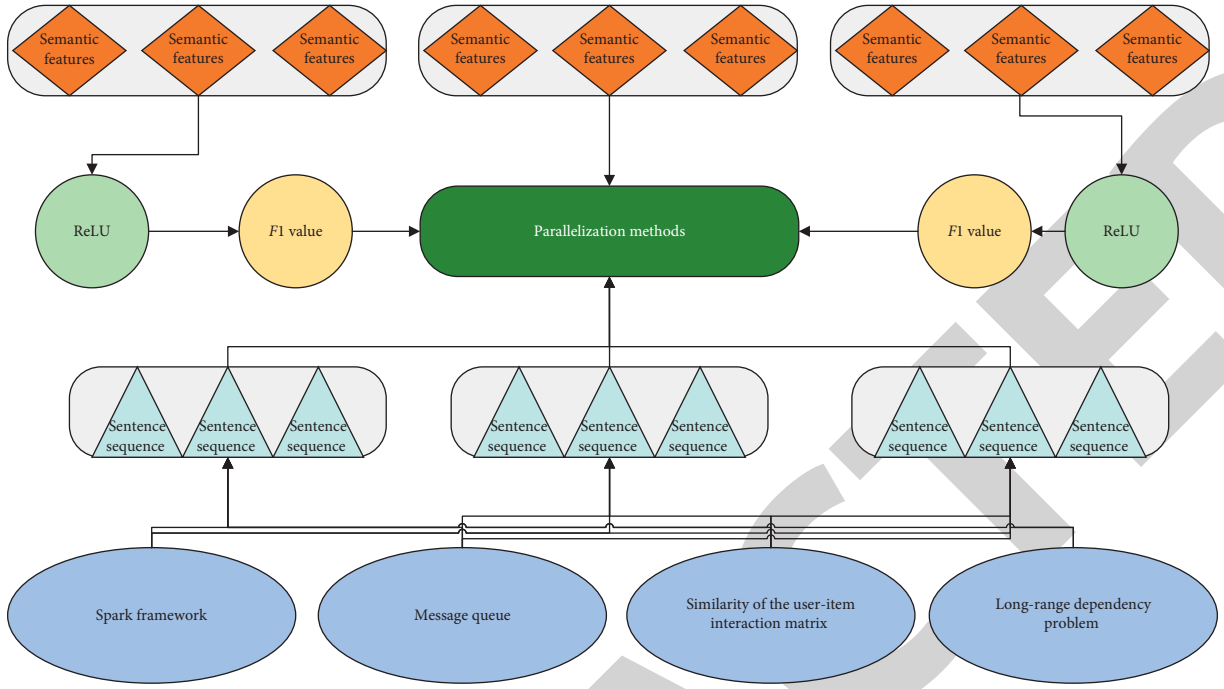

Figure 2: Path connection operation in phrase.

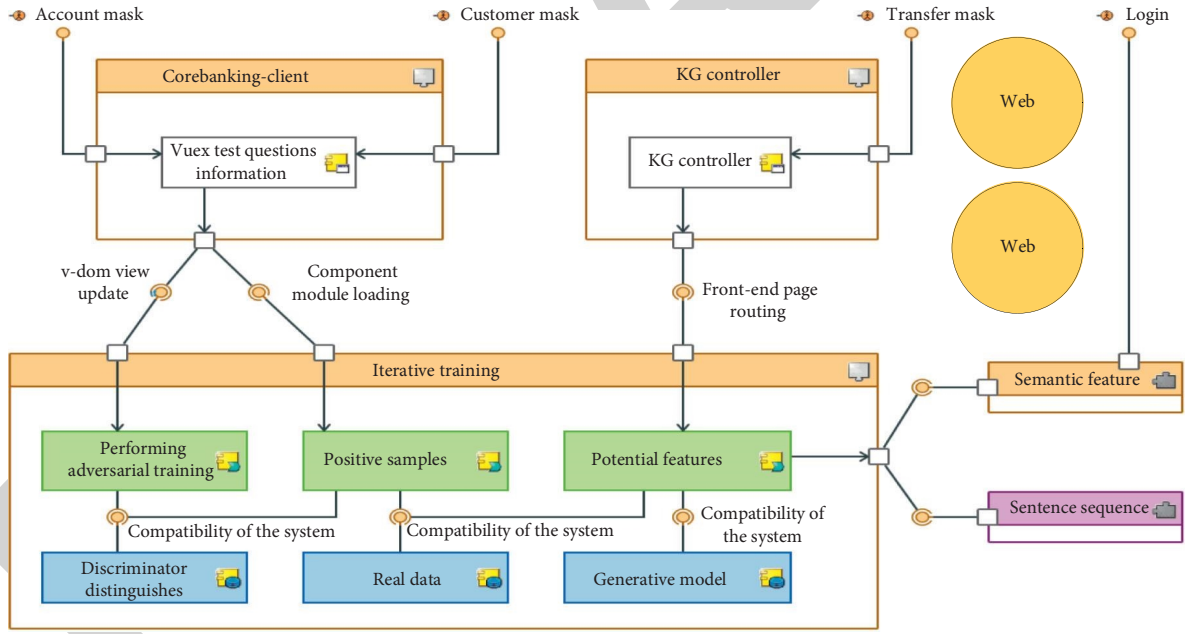

Figure 3: Vue logic structure diagram.

value. After adding lexical annotation features for each word in the threat intelligence sentence sequence, the model performance is improved by 0.36%, which shows that adding lexical annotation features can help improve the word-level features in the sentence sequence, and the effect is finally reflected in the relationship extraction results. It can also be seen that although the introduction of lexical annotation features does not contribute to the accuracy of the extraction results, it plays a key role in the improvement of the recall rate [20]. Further, after replacing the lexical annotation features with the position features of each word in the sentence sequence relative to the entity words, the model is retrained and the entity extraction results are statistically analysed, and it is found that the performance of the model is improved at both the accuracy and recall levels, resulting in an improvement in the overall performance of the model, as shown by a 1.21% increase in the $F1$ value. Finally, based on the acquisition of the lexical label features and the relative position features, the new and more complete input features are used as the input of the model, and the new and more

complete input features have a significant effect on improving the overall performance of the model. The additional lexical annotation features and relative location features added to the model design play an important role in improving the relationship extraction of threat intelligence entities, as shown in Figure 4.

Global feature extraction and local feature extraction of threat intelligence sentence sequences are considered, and the fusion of the two types of features is used as threat intelligence entity-relationship classification. In this section, the impact of global features, local features, and the fusion of the two on the performance of the model is verified. For the acquisition of global features, the BiGRU model is used; for the acquisition of local features, the PCNN model as described previously and the DGRU model in the model framework of this chapter are used, respectively, and the performance of the two types of models in acquiring semantic features of sentence sequences is evaluated.

We collected some entities in the geographic domain, 42,862 entities with 7,961 attributes, 19,564 entities with 1,730 attributes in Interactive Encyclopaedia, and 46,379 entities with 3,482 attributes in Wikipedia, of which 1,687 have a frequency greater than 1. In the experiment, although the attributes with the frequency of 1 occurrence account for a relatively large proportion, there is only one association available for this part of attributes, which is too specific to be used in this algorithm [21]. Therefore, we only selected attributes with frequencies greater than 1 for alignment; i.e., the 2,822 attributes in Encyclopaedia were aligned with 1,730 attributes in Interactive Encyclopaedia and 1,687 attributes in Wikipedia, respectively. Of course, before doing the attribute alignment, it is necessary to align the entities in the two knowledge graphs, and based on the results of entity alignment, the algorithm can be applied. We executed the entity alignment algorithm described in Chapter 3 on these two datasets and manually corrected some of the alignment results to ensure the high quality of the entity alignment results.

The similarity of attribute values is a very important factor for measuring the similarity of attributes. By extracting the values of all entities under different attributes to form a set of values, it is not difficult to find that the difference between attributes that do not agree on the set of attribute values is often obvious, and attributes that agree on different names tend to have some similarity in the set of values [22–28]. For example, for the attribute "place of birth," the elements in its value set are generally locations, while for "date of birth," its value is generally date-based data, and the difference between the two is obvious. For two attributes like "voice actor" and "CV," which are completely different in name, we can improve their similarity by calculating the degree of intersection of their value sets.

4. Results and Discussion

Text-based entity similarity association network construction: The experiment is divided into two parts: entity-chapter ID dictionary construction and association chapter discovery. To compare the algorithm efficiency, 10000, 20000,

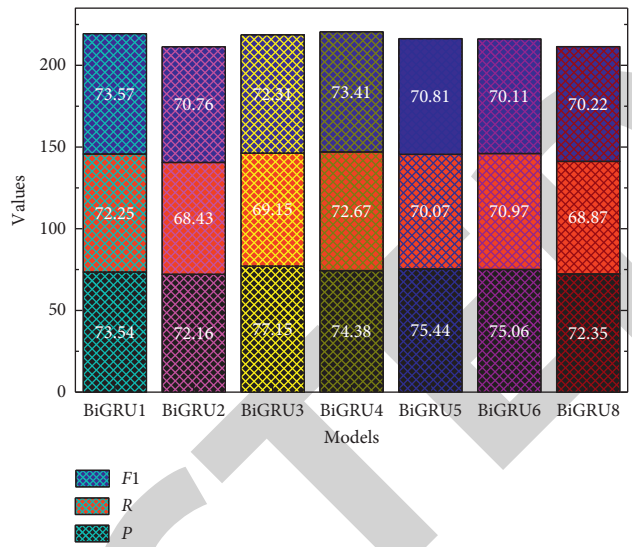

Figure 4: Effect of different input features on model performance.

and 30000 datasets in the association calculation method 1 are also used. Since the single-node dictionary construction and association calculation are more efficient in this dataset without running on the cluster, only the execution time of different datasets under a single node is compared, where the dictionary construction efficiency comparison experiment results are shown in Figure 5.

From the graph of the experimental results, it can be visually seen that the dictionary construction time increases with the increase of data volume. In the three datasets, the 20,000 data set is twice the 10,000 data set, the 30,000 data set is three times as long as the sub-10,000 data set, and the processing time for dictionary construction for the 20,000 data set is nearly two times as long as the 10,000 data set, and the 30,000 data set is nearly three times as long as the 10,000 data set. This shows that the dictionary construction time increases linearly with the increase of data volume in the dataset, which is because the dictionary can be obtained by traversing the dataset only once. For the chapter-to-chapter correlation calculation experiment, the dictionary is used to find the two correlated pairs of chapters in each dataset. For each dataset, the entity-chapter ID dictionary constructed by the above procedure is used to find the number of associated chapter pairs in the dataset for each chapter. The number of associated chapter pairs found using the entity-chapter ID dictionary indicates that this approach does not affect the results of the algorithm and is not listed here.

Figure 6 takes some samples from the set of high-frequency, medium-frequency, and low-frequency words in the word list and counts the changes in the number of times they are sampled when the negative sampling strategy is changed from NEG to NEG-TFIDF. The frequency of negatively sampled high-frequency words decreases significantly. The reduced number of sampled high-frequency words is distributed more evenly between medium-frequency words and low-frequency words with the same total training number. This proves that the NEG-TFIDF negative sampling strategy can bring more training times for the

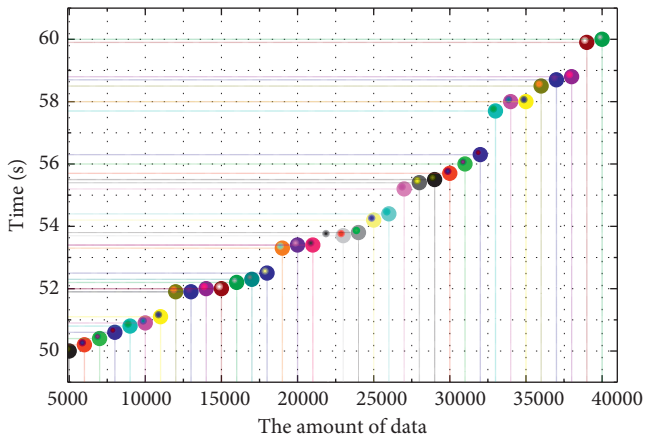

Figure 5: Comparison of time spent on building entity-chapter ID dictionaries with different datasets.

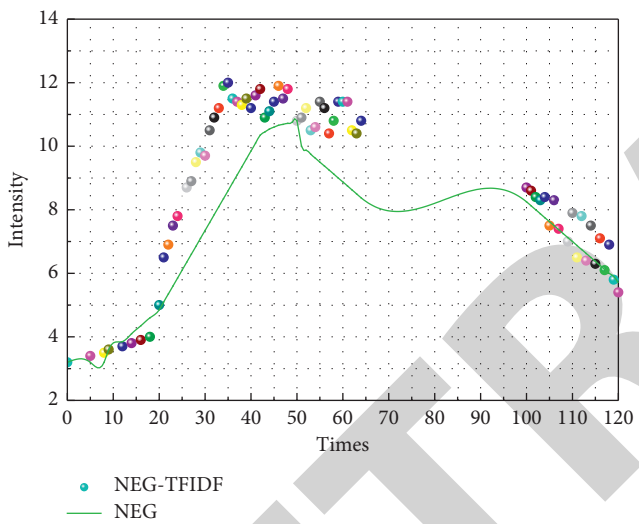

Figure 6: Comparison of the number of the word training.

medium-frequency words and low-frequency words, thus improving the word vector quality of these two types of words. In Figure 6, the experimental group is NEG-TFIDF and the control group is NEG.

Figure 7 explores the effect of the number of negative samples on the quality of the word vector training, and Word Analogy Overall Accuracy is used as the evaluation index. The trend of the curves in Figure 7 shows that, within a reasonable range, the difficulty of self-supervised training is increased by increasing the number of negative samples, which leads to the improvement of word vector quality; when the number of negative samples exceeds a certain threshold, the quality of NEG-based word vectors tends to decrease, which is the same as the results of Pennington's validation. The reason for this phenomenon is that an excessive number of negative samples raises the training of the word vectors to a relatively unreasonable difficulty, thus causing a decrease in the effect. In contrast, this problem is effectively improved when NEG-TFIDF is used as a negative sampling strategy. The reason for this is that NEG-TFIDF presents a more reasonable sampling strategy, while the unreasonableness of NEG sampling is amplified when the number of sampled samples increases, thus affecting the quality of the word vectors.

Querying or updating the relational database through the D2RQ tool requires two translation mappings, resulting in insufficient performance. Therefore, in this paper, the Neo4j graph database is selected for knowledge storage to query performance and facilitate the graphical presentation of data. Neo4j mainly consists of two basic data types (nodes and edges), where nodes represent the entities of the knowledge graph and edges store the relationships between entities. Encoder1 acts as a global information provider, while Encoder2 acts as a local feature extractor and feeds directly into the classifier. Also, two models are designed to interact with each other. With the awareness of global information, the authors' proposed method is better able to learn instance-specific local features, thus avoiding complex upper-level operations. Experiments conducted on eight benchmark datasets show that the architecture largely facilitates the development of local feature-driven models and outperforms previous best models in a fully supervised setting. Therefore, the Neo4j graph database constructed in this paper is designed as follows: corresponding to the four concepts of the ontology library, there are four types of nodes: Movie, Role, Distributor, and Genre; corresponding to the object properties of the ontology library, there are seven types of edges, director, producer-distributor, and cinematography, etc.; corresponding to the 11 data attributes of the ontology, the node attributes are sole birthplace, distributor name, movie language, and movie release time, etc. Based on the above design, this section finally stores a series of knowledge such as entities and relationships to Neo4j, and Figure 8, for example, shows the knowledge subgraph of 50 Romance class movies.

With the development of information technology and the Internet industry, information overload has become a problem for people to handle information. The birth of recommendation systems has greatly alleviated this difficulty. Considering that the traditional collaborative filtering recommendation algorithm only takes the user-item interaction matrix as input data, which is prone to sparsity and cold-start problems, this paper focuses on a personalized recommendation scheme based on knowledge graphs, hoping to use the rich semantic information of items in knowledge graphs to strengthen the connection between users and items and bring additional diversity and interpretation to the recommendation algorithm. To realize the personalized recommendation scheme based on knowledge graphs, this paper investigates the distributed representation learning algorithm of knowledge graphs and the design and implementation of personalized recommendation algorithms. For the representation learning of knowledge graphs, this paper points out that traditional graph distributed representation methods lose higher-order similarity at the subgraph level. To this end, this paper proposes an RNN-based knowledge graph distributed representation model KG-GRU, which models subgraph similarity using multiple paths containing entities and relations and represents relations and entities in the same embedding vector space.

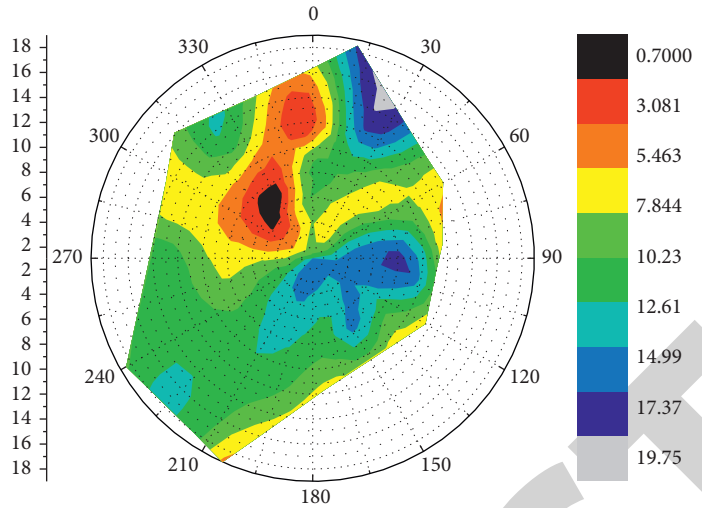

Figure 7: Effect of the number of negative samples on the quality of word vectors.

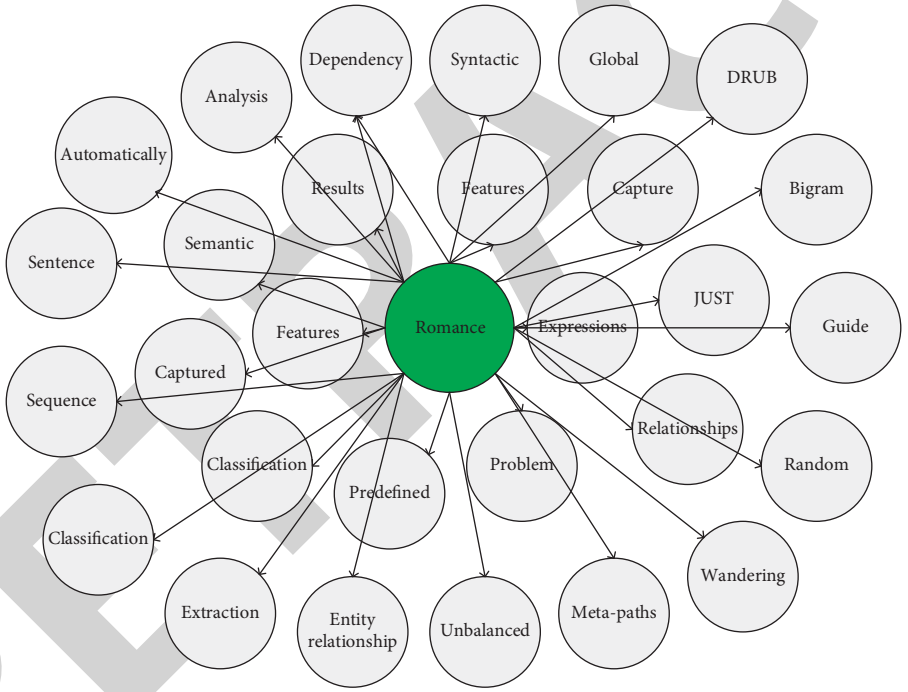

Figure 8: Entity example diagram.

Also, this paper proposes a skip or dwell strategy just to guide random wandering for data sampling of the knowledge graph, avoiding the problems of manually constructing meta-paths and an unbalanced distribution of entity types.

The entity-relationship extraction problem is modelled as a classification task of predefined relationships. In this process, to obtain complete sentence-level semantic features, this chapter proposes using the Bigram model and DRUB model to capture global features and local features, respectively and fusing the two types of features to obtain unified sentence-level semantic features. Based on this, the attention mechanism based on syntactic dependency is proposed, and the distance between words in the threat intelligence sentence sequence is defined by the syntactic dependency analysis results, and the dependent attention score of semantic features is obtained by combining the self-attention calculation process, which is applied to the semantic features captured automatically in the first stage to obtain the final feature expression of the sentence sequence as the input features for relationship classification. Also, to enhance the semantic feature expressions, this chapter designs a semantic enhancement adversarial learning framework based on generative adversarial networks, in which the automatically captured features are used as negative samples, and the samples obtained from the syntactic dependency analysis combined with manual processing are used as positive samples, between which adversarial learning is performed so that the automatically captured features of the

model are enhanced and the semantic relationships among threat intelligence entities are classified with complete feature expressions. The experimental results demonstrate the effectiveness of the proposed method in this chapter.

5. Conclusions

In this paper, the current parallelization techniques and knowledge graph construction techniques are studied. Combining event-oriented knowledge graph construction and chapter-oriented knowledge graph construction, the process of knowledge graph construction and the algorithms involved in each process are analysed, the characteristics of each algorithm are analysed, and various parallelization methods are designed and applied according to the characteristics of the algorithms. In this paper, the knowledge graph construction is mainly divided into four stages: data acquisition, knowledge extraction, knowledge representation, and knowledge processing. A distributed data acquisition architecture is designed for the data acquisition phase, and since the architecture uses message queues as message middleware, the number of nodes can be flexibly configured during data acquisition, and data acquisition can be efficiently carried out under multiple nodes; in the knowledge extraction phase, entity extraction and relationship extraction methods are mainly analysed, and three parallelization methods are designed for knowledge extraction. In the knowledge processing stage, we mainly analyse the algorithms related to cooccurrence relationship discovery, realize the parallelization of association network construction and hierarchical clustering fusion algorithm, and design a more efficient method for the calculation of associate degrees based on text entities. The experiments show that the parallelized cooccurrence relationship discovery algorithm proposed in this paper can significantly reduce processing time compared with a single node.

Data Availability

Data sharing is not applicable to this article as no datasets were generated or analysed during the current study.

Conflicts of Interest

The authors declare that there are no conflicts of interest.

Acknowledgments

This work was supported by Henan Soft Science Research Program Project (212400410192), Research on the Application of Recommendation Algorithm Based on Multivariate Collaborative Filtering in Medical Practice Qualification Examination.

References

[1] X. Wang, D. Wang, C. Xu, X. He, Y. Cao, and T.-S. Chua, "Explainable reasoning over knowledge graphs for recommendation," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 1, pp. 5329–5336, Honolulu, HI, USA, February 2019.

[2] P. Qin, X. Wang, W. Chen, C. Zhang, W. Xu, and W. Y. Wang, "Generative adversarial zero-shot relational learning for knowledge graphs," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 5, pp. 8673–8680, New York, NY, USA, February 2020.

[3] S. Guan, X. Jin, Y. Wang et al., "Self-learning and embedding based entity alignment," *Knowledge and Information Systems*, vol. 59, no. 2, pp. 361–386, 2019.

[4] W. Liu, P. Zhou, Z. Zhao et al., "K-bert: enabling language representation with knowledge graph," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 3, pp. 2901–2908, New York, NY, USA, February 2020.

[5] S. K. Mohamed, V Nováček, and N. Aayah, "Discovering protein drug targets using knowledge graph embeddings," *Bioinformatics*, vol. 36, no. 2, pp. 603–610, 2020.

[6] D. Reker, P. Schneider, G. Schneider, and J. B. Brown, "Active learning for computational chemogenomics," *Future Medicinal Chemistry*, vol. 9, no. 4, pp. 381–402, 2017.

[7] Y. Chen, T. A. Lask, Q. Mei et al., "An active learning-enabled annotation system for clinical named entity recognition," *BMC Medical Informatics and Decision Making*, vol. 17, no. 2, pp. 35–44, 2017.

[8] N. Noy, Y. Gao, A. Jain, A. Narayanan, A. Patterson, and J. Taylor, "Industry-scale knowledge graphs: lessons and challenges: five diverse technology companies show how it's done," *Queue*, vol. 17, no. 2, pp. 48–75, 2019.

[9] S. Amreen, A. Mockus, Z. Russell, C. Bogart, and Y. Zhang, "ALFAA: active learning fingerprint based anti-aliasing for correcting developer identity errors in version control systems," *Empirical Software Engineering*, vol. 25, no. 2, pp. 1136–1167, 2020.

[10] M. Gaur, K. Faldu, and S. Amit, "Semantics of the black-box: can knowledge graphs help make deep learning systems more interpretable and explainable?" *IEEE Internet Computing*, vol. 25, no. 1, pp. 51–59, 2021.

[11] V. Unnikrishnan, C. Beyer, P. Matuszyk et al., "Entity-level stream classification: exploiting entity similarity to label the future observations referring to an entity," *International Journal of Data Science and Analytics*, vol. 9, no. 1, pp. 1–15, 2020.

[12] M. Zhou, N. Duan, S. Liu, and H.-Y. Shum, "Progress in neural NLP: modeling, learning, and reasoning," *Engineering*, vol. 6, no. 3, pp. 275–290, 2020.

[13] A. Rossi, D. Barbosa, D. Firmani, A. Matinata, and P. Merialdo, "Knowledge graph embedding for link prediction: a comparative analysis," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 15, no. 2, pp. 1–49, 2021.

[14] R. Celebi, E. Y. HuseyinUyar, O. D. OzgurGumus, and M. Dumontier, "Evaluation of knowledge graph embedding approaches for drug-drug interaction prediction in realistic settings," *BMC Bioinformatics*, vol. 20, no. 1, pp. 1–14, 2019.

[15] K. Morton, P. Wang, C. Bizon et al., "ROBOKOP: an abstraction layer and user interface for knowledge graphs to support question answering," *Bioinformatics*, vol. 35, no. 24, pp. 5382–5384, 2019.

[16] Q. Zhang, J. Lu, D. Wu, and G. Zhang, "A cross-domain recommender system with kernel-induced knowledge transfer for overlapping entities," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 7, pp. 1998–2012, 2018.

[17] H. Jin, C. Li, J. Zhang, L. Hou, J. Li, and P. Zhang, "XLORE2: large-scale cross-lingual knowledge graph construction and application," *Data Intelligence*, vol. 1, no. 1, pp. 77–98, 2019.
[18] K. Kucher, C. Paradis, M. Sahlgren, and A. Kerren, "Active learning and visual analytics for stance classification with ALVA," *ACM Transactions on Interactive Intelligent Systems (TiiS)*, vol. 7, no. 3, pp. 1–31, 2017.
[19] N. Kertkeidkachorn and R. Ichise, "An automatic knowledge graph creation framework from natural language text," *IEICE Transactions on Information and Systems*, vol. 101, no. 1, pp. 90–98, 2018.
[20] J. Yang, Y. Zhao, J. Liu et al., "No reference quality assessment for screen content images using stacked autoencoders in pictorial and textual regions," *IEEE Transactions on Cybernetics*, pp. 1–13, 2020.
[21] F. Z. Smaili, X. Gao, and R. Hoehndorf, "OPA2Vec: combining formal and informal content of biomedical ontologies to improve similarity-based prediction," *Bioinformatics*, vol. 35, no. 12, pp. 2133–2140, 2019.
[22] P. Lin, S. Qi, and Y. Wu, "Fact checking in knowledge graphs with ontological subgraph patterns," *Data Science and Engineering*, vol. 3, no. 4, pp. 341–358, 2018.
[23] L. Ding, S. Li, H. Gao, C. Chen, and Z. Deng, "Adaptive partial reinforcement learning neural network-based tracking control for wheeled mobile robotic systems," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 50, no. 7, pp. 2512–2523, 2018.
[24] W. Wang, Z. Gong, J. Ren, F. Xia, Z. Lv, and W. Wei, "Venue topic model–enhanced joint graph modelling for citation recommendation in scholarly big data," *ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP)*, vol. 20, no. 1, pp. 1–15, 2020.
[25] J. Zhang and G. Qu, "Physical unclonable function-based key sharing via machine learning for IoT security," *IEEE Transactions on Industrial Electronics*, vol. 67, no. 8, pp. 7025–7033, 2019.
[26] C. Chen, K. Li, W. Wei, J. T. Zhou, and Z. Zeng, "Hierarchical graph neural networks for few-shot learning," *IEEE Transactions on Circuits and Systems for Video Technology*, p. 1, 2020.
[27] J. Yang, C. Wang, H. Wang, and Q. Li, "A RGB-D based real-time multiple object detection and ranging system for autonomous driving," *IEEE Sensors Journal*, vol. 20, no. 20, pp. 11959–11966, 2020.
[28] I. Abdelaziz, A. Fokoue, and O. Hassanzadeh, P. Zhang and M. Sadoghi, Large-scale structural and textual similarity-based mining of knowledge graph to predict drug–drug interactions," *Journal of Web Semantics*, vol. 44, pp. 104–117, 2017.